LEVEL II

NRL Memorandum Report 4414

A FORTRAN Computer Program for Calculating the Prolate Spheroidal Angular Functions of the First Kind.

B. J. Katz and A. L. Van Buren

Underwater Sound Reference Detachment
P.O. Box 8337
Orlando, FL 32856

NRL-MR-4414

13 Mar 1981

43

RR011D8

RR011D842

NAVAL RESEARCH LABORATORY
Washington, D.C.

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER: NRL Memorandum Report 4414 ✓	2. GOVT ACCESSION NO.: AD-A095 922
3. RECIPIENT'S CATALOG NUMBER	
4. TITLE (and Subtitle): A FORTRAN COMPUTER PROGRAM FOR CALCULATING THE PROLATE SPHEROIDAL ANGULAR FUNCTIONS OF THE FIRST KIND	5. TYPE OF REPORT & PERIOD COVERED: Interim report on a continuing NRL problem.
	6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s): B. J. Patz* and A. L. Van Buren	8. CONTRACT OR GRANT NUMBER(s)
9. PERFORMING ORGANIZATION NAME AND ADDRESS: Underwater Sound Reference Detachment, Naval Research Laboratory, P. O. Box 8337, Orlando, FL. 32856	10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS: 61153N; RR-011-08-42; 0589-0-1
11. CONTROLLING OFFICE NAME AND ADDRESS: ONR to Naval Research Laboratory, Systems Research & Technology Directorate, Washington, D. C. 20375	12. REPORT DATE: March 13, 1981
	13. NUMBER OF PAGES: 42
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office)	15. SECURITY CLASS. (of this report): UNCLASSIFIED
	15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

*Mr. Patz, a student at Rensselear Polytechnic Institute, was at the USRD under the 1980 Summer Employment Program.

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Spheroidal wave functions
Prolate spheroids
Computer programs
Helmholtz wave equation

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

The Helmholtz wave equation $(\nabla^2+k^2)\,\psi = 0$ is separable in prolate spheroidal coordinates (η, ξ, ϕ) with the solution $\psi = R(c, \xi)S(c, \eta)\Phi(\phi)$. Here $c = ka/2$, where a is the interfocal distance. A computer program called PANGFN has been developed to numerically evaluate in double-precision arithmetic the angular function of the first kind $S_{m\ell}^{(1)}(c, \eta)$ for any desired values of c, m, ℓ, and the argument $\theta = \cos^{-1}\eta$. The program is written in universal FORTRAN and should run on any computer that accepts this language. Special techniques are used to avoid overflow and underflow problems. By the use of logarithms, PANGFN can compute the angular function even when its value exceeds the exponent range of the computer. This report describes the —
(Continues)

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE S/N 0102-LF-014-6601

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

20. Abstract (Continued)

principle features of PANGFN. Included are discussions of the significant FORTRAN variable names, dimensions and storage, parameter input, parameter ranges, computational procedure, computation time, printed output, and accuracy of the results. A sample output and a computer listing of LINPRO are attached as appendices.

CONTENTS

A FORTRAN COMPUTER PROGRAM FOR CALCULATING THE PROLATE SPHEROIDAL ANGULAR FUNCTIONS OF THE FIRST KIND

INTRODUCTION

The Helmholtz or scalar wave equation for steady waves $(\nabla^2 + k^2)\psi = 0$, where $k = 2\pi/\lambda$ with λ equal to the wavelength, is separable in prolate spheroidal coordinates (ξ, η, ϕ). The factored solution is written as $R(c,\xi)S(c,\eta)\Phi(\phi)$. Here $c = ka/2$, where a is the interfocal distance of the elliptic cross section of the spheroid.

The angular function of the first kind $S_{m\ell}^{(1)}(c,\eta)$ is one of two independent solutions to the ordinary differential equation in the angle coordinate η arising from the separation of variables. This solution is characterized by the four parameters m, ℓ, c, η. For each of the choices of m, ℓ, c, and η there exists a set of solutions to the prolate angular equation, each solution characterized by a separation constant or eigenvalue $A_{m\ell}$. As with the corresponding associated Legendre functions of spherical geometry, it is often convenient to specify the argument η in terms of the angle $\theta = \cos^{-1}\eta$; i.e., $S_{m\ell}^{(1)}(c,\eta) \equiv S_{m\ell}^{(1)}(c,\cos\theta)$.

The computer program PANGFN calculates numerical values for the angular function of the first kind $S_{m\ell}^{(1)}(c,\eta)$ and the associated eigenvalues $A_{m\ell}$ for desired values of m, ℓ, c, and θ. PANGFN is intended to replace the prolate portion of the FORTRAN computer program ANGLFN [1], which was previously developed at the Naval Research Laboratory (NRL) to evaluate both the prolate and the oblate angular functions of the first kind. Unfortunately, ANGLFN and two companion computer programs for calculating the prolate and oblate radial functions of both kinds and their first derivatives [2,3] were developed around the large exponent size (±307) of the CDC3800 computer at NRL. These programs are not easily modified to run on computers with a significantly smaller exponent range. The program PANGFN, however, is designed to run on computers with any exponent range. It is written in universal FORTRAN and should run on any computer that accepts this language.

Manuscript submitted December 10, 1980.

Similar universal computer programs will be developed in the future for the oblate angular functions, prolate radial functions, and oblate radial functions. A universal program called LINPRO [4] already exists for calculating the linear prolate functions and eigenvalues. These functions, which are useful in the representation of band-limited and time-limited physical processes, are constructed from the prolate angular functions with m set equal to zero.

ANALYSIS

The prolate spheroidal system can be formed by rotating the two-dimensional elliptic coordinate system, consisting of ellipses and hyperbolas, about the major axis of the ellipse. The prolate spheroidal coordinates (ξ, η, ϕ) with $1 \le \xi \le \infty$, $-1 \le \eta \le 1$, and $0 \le \phi \le 2\pi$ are related to the Cartesian coordinates (x,y,z) by the transformations:

$$x = (a/2)[(\xi^2 - 1)(1 - \eta^2)]^{\frac{1}{2}}\cos\phi \quad , \tag{1}$$

$$y = (a/2)[(\xi^2 - 1)(1 - \eta^2)]^{\frac{1}{2}}\sin\phi \quad , \tag{2}$$

and

$$z = a\xi\eta/2 \quad . \tag{3}$$

The spheroidal coordinates ξ = const., η = const., and ϕ = const. define the following set of orthogonal surfaces, as shown in Fig. 1:

ellipsoid of revolution: $$\frac{x^2 + y^2}{(a/2)^2(\xi^2-1)} + \frac{z^2}{(a/2)^2\xi^2} = 1 \quad , \tag{4}$$

hyperbolcid of two sheets: $$\frac{x^2 + y^2}{(a/2)^2(1-\eta^2)} - \frac{z^2}{(a/2)^2\eta^2} = -1 \quad , \tag{5}$$

and

half plane containing the z-axis: $$\phi = \tan^{-1}(y/x) \quad . \tag{6}$$

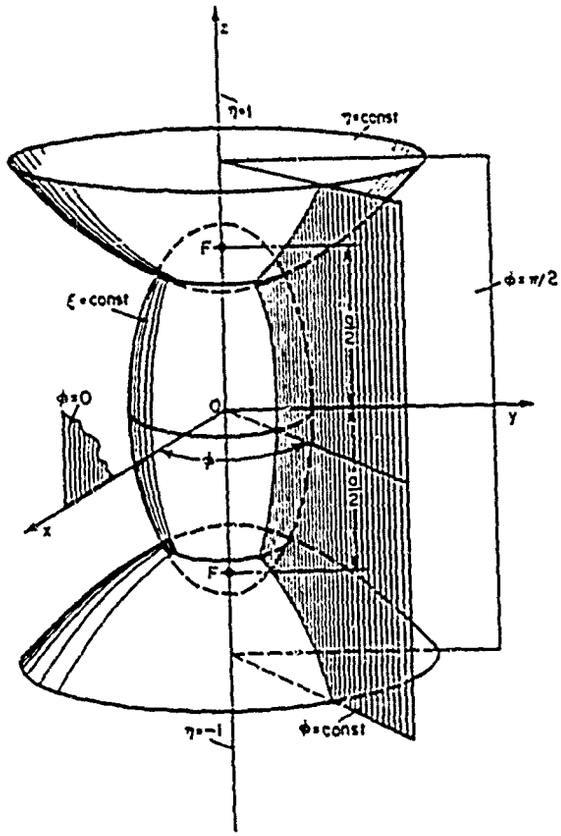

Fig. 1 - The prolate spheroidal coordinate system

Separation of the Helmholtz equation

$$(\nabla^2 + k^2)\,\psi = 0 \tag{7}$$

in prolate spheroidal coordinates yields the following solution:

$$\psi_{m\ell} = R_{m\ell}(c,\xi)S_{m\ell}(c,\eta)\Phi_m(\phi)\ , \tag{8}$$

where $R_{m\ell}(c,\xi)$ is the radial function, $S_{m\ell}(c,\eta)$ is the angular function, and $\Phi_m(\phi)$ is the azimuthal function.

The functions $R_{m\ell}(c,\xi)$, $S_{m\ell}(c,\eta)$, and $\Phi_m(\phi)$ satisfy the following ordinary differential equations:

$$\frac{d}{d\xi}[(\xi^2-1)\frac{dR_{m\ell}}{d\xi}] - [A_{m\ell} - c^2\xi^2 + \frac{m^2}{\xi^2-1}]R_{m\ell} = 0\ , \tag{9}$$

$$\frac{d}{d\eta}[(1-\eta^2)\frac{dS_{m\ell}}{d\eta}] + [A_{m\ell} - c^2\eta^2 - \frac{m^2}{1-\eta^2}]S_{m\ell} = 0 \ , \qquad (10)$$

$$\frac{d^2\phi_m}{d\phi^2} + m^2\phi_m = 0 \ , \qquad (11)$$

where m and $A_{m\ell}$ are the two separation constants or eigenvalues occuring in the separation of variables. Throughout the rest of this manuscript the term eigenvalue is understood to be $A_{m\ell}$, unless otherwise stated.

In physical problems in which the field has to be periodic and unique over the range of the azimuthal coordinate ϕ, it is required that m be an integer. For fixed m and $c \neq 0$ the numbers $A_{m\ell}$ for which Eqs. (9) and (10) have non-trivial convergent solutions are ordered numerically in an ascending series and labeled with the integers $\ell = m, m+1, m+2, \ldots$. When $c \longrightarrow 0$, Eq. (10) reduces to the ordinary differential equation of the associated Legendre functions whose eigenvalues $A_{m\ell}$ are equal to $\ell(\ell+1)$.

The two independent solutions of Eq. (9) are known as the radial function of the first kind $R^{(1)}_{m\ell}(c,\xi)$ and the radial function of the second kind $R^{(2)}_{m\ell}(c,\xi)$. Similarly, the solutions of Eq. (10) are known as the angular function of the first kind $S^{(1)}_{m\ell}(c,\eta)$ and the angular function of the second kind $S^{(2)}_{m\ell}(c,\eta)$. Excellent discussions of the uses and properties of these functions are found in the monographs by Meixner and Schäfke [5] and Flammer [6].

Three volumes of tables of numerical values for the prolate radial functions and their first derivatives with respect to ξ were published in 1970 [7]. These volumes contain entries for the following range of parameter: m = 0 (Volume 1), m = 1 (Volume 2), m = 2 (Volume 3), ℓ = m, m+1, ..., m+49; ξ = 1.00000001, 1.0000001, ..., 1.01, 1.02 (0.02) 1.2*, 1.4 (0.2) 2.0, 4.0 (2.0) 10.0; c = 0.1 (0.1) 1.0, 2.0 (1.0) 10.0, 12.0 (2.0) 30.0, 35.0, 40.0. A single volume of tables of numerical values for the prolate angular functions and the linear prolate eigenvalues was published in 1975 [8]. The range of variables covered in this volume is m = 0; ℓ = 0 (1) 49; θ = 0° (1°) 90°, where $\eta = \cos\theta$; c = 0.1 (0.1) 1.0, 2.0 (1.0) 10.0, 12.0 (2.0) 30.0, 35.0,

---

*The notation 1.02 (0.02) 1.2 indicates 1.02, 1.04, 1.06, ..., 1.2.

40.0. Three volumes of oblate radial functions [9] and one volume of oblate angular functions [10] were also published.

PROLATE ANGULAR FUNCTION OF THE FIRST KIND

Since the angular function of the first kind $S_{m\ell}^{(1)}(c,\eta)$ reduces to the associated Legendre function of the first kind $P_{\ell}^{m}(\eta)$ in the limit $c \longrightarrow 0$, it is convenient to expand the angular function in the following series:

$$S_{m\ell}^{(1)}(c,\eta) = \sum_{n=0,1}^{\infty}{}' d_n(c|m\ell) P_{m+n}^{m}(\eta) \ . \tag{12}$$

The expansion coefficients $d_n(c|m\ell)$ are sometimes called the D constants. The prime sign on the sum indicates that $n = 0, 2, 4, \ldots$, if $\ell-m$ is even, or $n = 1, 3, 5, \ldots$, if $\ell-m$ is odd. This ensures that the angular function $S_{m\ell}^{(1)}(c,\eta)$ has the same evenness or oddness with respect to η as the corresponding associated Legendre function $P_{\ell}^{m}(\eta)$, i.e., $S_{m\ell}^{(1)}(c,\eta)$ is an even function of η if $\ell-m$ is even and is an odd function if $\ell-m$ is odd. Ferrers' [11] definition of the associated Legendre functions has been adopted. Thus,

$$P_n^m(\eta) = (1-\eta^2)^{m/2} \frac{d^m P_n(\eta)}{d\eta^m} \ . \tag{13}$$

This leads to the following recursion relation:

$$(n-m+1)P_{n+1}^{m}(\eta) = (2n+1)\eta P_n^m(\eta) - (n+m)P_{n-1}^{m}(\eta) \ , \tag{14}$$

where

$$P_{m-1}^{m}(\eta) = 0 \ , \tag{15}$$

$$P_m^m(\eta) = (2m-1)!!(1-\eta^2)^{m/2} \ , \tag{16}$$

with $(2m-1)!! = (2m-1)(2m-3)\ldots(3)(1)$.

Substitution of Eq. (12) into Eq. (10) and use of the above recursion relation for the Legendre functions leads to the following three-term recursion relation in terms of the D constants.

$$\frac{(2m+n+2)(2m+n+1)}{(2m+2n+3)(2m+2n+5)}\, c^2 d_{n+2}$$

$$+ \left[(m+n)(m+n+1) - A_{m\ell} + \frac{2(m+n)(m+n+1)-2m^2-1}{(2m+2n+3)(2m+2n-1)}\, c^2\right] d_n \qquad (17)$$

$$+ \frac{n(n-1)}{(2m+2n-3)(2m+2n-1)}\, c^2 d_{n-2} = 0 \ .$$

Equation (10) has two regular singularities at $\eta = \pm 1$. If the choice of $A_{m\ell}$ is arbitrary, solutions given by Eq. (12) may be divergent at either $\eta = 1$ or $\eta = -1$. It is necessary in physical applications, however, that $S^{(1)}_{m\ell}(c,\eta)$ be finite at both $\eta = 1$ and $\eta = -1$. It is also required that Eq. (12) converge. This requires that $d_{n+2}/d_n \longrightarrow 0$ as $n \longrightarrow \infty$. Use of eigenvalues that satisfy both this condition and Eq. (17) above will result in successful expansions of $S^{(1)}_{m\ell}(c,\eta)$.

The desired eigenvalues are obtained by a two-step process. First an approximation to the eigenvalue is calculated by use of formulas such as a polynomial expansion in c. With this approximation as a starting value, the eigenvalue is then calculated using a variational procedure developed by Bouwkamp [12]. In the Bouwkamp procedure the three-term recursion relation found in Eq. (17) is rewritten in the following two forms:

$$N^m_n = \frac{-\beta^m_n}{\gamma^m_n - A_{m\ell} + N^m_{n+2}} \ , \ n \geq 2 \ , \qquad (18)$$

and

$$N^m_{n+2} = -\gamma^m_n + A_{m\ell} - \frac{\beta^m_n}{N^m_n} \ , \ n \geq 2 \ , \qquad (19)$$

where

$$\gamma_n^m = (m+n)(m+n+1) + (c^2/2)\left[1 - \frac{4m^2-1}{(2m+2n-1)(2m+2n+3)}\right] , \quad n \geq 0 , \quad (20)$$

$$\beta_n^m = \frac{n(n-1)(2m+n)(2m+n-1)c^4}{(2m+2n-1)^2(2m+2n-3)(2m+2n+1)} , \quad n \geq 2 , \quad (21)$$

$$N_n^m = \frac{(2m+n)(2m+n-1)}{(2m+2n-1)(2m+2n+1)} c^2 \frac{d_n}{d_{n-2}} , \quad n \geq 2 . \quad (22)$$

Requiring that $d_{n+2}/d_n \longrightarrow 0$ as $n \longrightarrow \infty$ is equivalent to requiring that $N_n^m \longrightarrow 0$ as $n \longrightarrow \infty$. Equation (18) can be rewritten as a continued fraction in terms of γ_n^m, γ_{n+2}^m, ..., β_n^m, β_{n+2}^m, ..., and $A_{m\ell}$. Equation (19) can also be rewritten as a continued fraction in terms of γ_{n-2}^m, γ_{n-4}^m, ..., β_{n-2}^m, β_{n-4}^m, ..., and $A_{m\ell}$, using the fact that $N_2^m = -\gamma_0^m \div A_{m\ell}$ and $N_3^m = -\gamma_1^m + A_{m\ell}$ to terminate the sequence.

The starting value for the eigenvalue $A_{m\ell}$ is inserted into the two continued fractions, one with diminishing subscripts and one with increasing subscripts. $N_{\ell-m+2}^m$ is calculated using both expressions, and the difference in the two values is used to determine a correction to $A_{m\ell}$. The process is repeated until $A_{m\ell}$ is obtained to the desired degree of accuracy.

When the value of $A_{m\ell}$ has been accurately determined, the D constants can be calculated by successive application of Eq. (17). The D constants obtained from Eq. (17) are un-normalized since this equation is homogeneous. A suitable normalization is established by requiring that the angular functions have the same normalization as the associated Legendre functions. Thus

$$\int_{-1}^{1}\left[S_{m\ell}^{(1)}(c,\eta)\right]^2 d\eta = \int_{-1}^{1}\left[P_\ell^m(\eta)\right]^2 d\eta = \frac{2(\ell+m)!}{(2\ell+1)(\ell-m)!} . \quad (23)$$

Substituting the expansion for $S_{m\ell}^{(1)}(c,\eta)$ into Eq. (23) and using the known orthogonal properties of the Legendre functions provides the following

normalizing relation for the D constants:

$$2\sum_{n=0,1}^{\infty}{}' \frac{(n+2m)!\ d_n^2}{(2m+2n+1)\ n!} = \frac{2(\ell+m)!}{(2\ell+1)(\ell-m)!} \quad . \tag{24}$$

This normalization scheme was first used by Meixner and Schäfke [5]. It has the practical advantage of eliminating the need to numerically evaluate the normalization factor $\int_{-1}^{1}[S_{m\ell}^{(1)}(c,\eta)]^2 d\eta$ which is often encountered in problems involving expansions in angular functions. Once the D constants have been normalized, the angular function can be evaluated using Eq. (12). Alternatively, the angular function can be evaluated using un-normalized D constants and then corrected by multiplying by the ratio of the normalized to un-normalized value of any one of the D constants, say $d_{\ell-m}$.

BRIEF DESCRIPTION OF PANGFN

The computer program PANGFN calculates, in double precision arithmetic, numerical values for $A_{m\ell}$ and $S_{m\ell}^{(1)}(c,\eta)$ for desired values of m, ℓ, c, and θ, where $\theta = \cos^{-1}(\eta)$. The program is written in universal FORTRAN and should run on any machine that accepts the language. The main program calls four subroutines: "PLEG", "GETEIG", "CONVER", and "OUTPUT". "PLEG" generates the associated Legendre functions; "GETEIG" produces an approximation to the eigenvalue for starting the Bouwkamp procedure; "CONVER" uses the Bouwkamp procedure to determine the eigenvalue; and "OUTPUT" prints the final results. The processes of these routines are described in more detail in the section entitled COMPUTATIONAL PROCEDURES.

PANGFN is designed to accommodate the exponent range and word length of the user's computer. It is necessary to change the first two executable statements in the program to correspond to the user's computer. These statements specify NDEC, the number of decimal digits available for double precision numbers, and NEX, the maximum possible exponent. In the unlikely event that NDEC exceeds 36, the value for π given by PI in the third executable statement must be extended to NDEC digits. The user should also set the array dimensions large enough to accommodate the desired parameter ranges, as described in the section entitled DIMENSIONS AND STORAGE.

The remainder of this report describes the program PANGFN. Included are a listing of the significant FORTRAN variables and a description of the major computational blocks. A discussion of parameter input and resulting output follows. A listing of the program and a sample output are also given.

SIGNIFICANT FORTRAN VARIABLE NAMES

ARG: Value of the angle θ for which the angular function $S_{m\ell}^{(1)}(c,\cos\theta)$ is calculated.

ARGG: Temporary holding array for outputing ARG.

ARG1: Input parameter and initial value of the angle θ for which the angular function $S_{m\ell}^{(1)}(c,\cos\theta)$ is desired.

BLIST: Array used in the Bouwkamp procedure. It contains the β_n^m coefficients defined in Eq. (21).

C: c.

C1: Input parameter and initial value of c.

CL: Eigenvalue $A_{m\ell}$. Initially equal to the approximation returned by subroutine "GETEIG", then converges to $A_{m\ell}$ during Bouwkamp procedure.

CLSPAC: The estimated spacing between two eigenvalues. Used to determine if the Bouwkamp procedure has converged to the proper eigenvalue and to approximate an upper bound on a new estimate if a new starting value is needed.

COEF: Term used in computing the normalizing factor.

CSQ: c^2.

DARG: Input parameter and step size used to generate ARG.

DC: Input parameter and step size used to generate c.

DEC: A constant set equal to $10^{-(NDEC+1)}$. Used to determine convergence of the angular function series of Eq. (12).

DEC2: A constant set equal to $10^{-(NDEC-1)}$. Used to determine convergence of the Bouwkamp eigenvalue procedure.

DNUM: The normalizing factor used to provide Meixner-Schäfke normalization for the angular functions. It is equal to the normalized value of $d_{\ell-m}$.

EIG2, EIG3, and EIG4: Previous eigenvalues used to generate the eigenvalue estimate when ℓ-m is large enough.

ENR: The array used to contain the D constant ratios returned from "CONVER". The coefficients N_n^m are calculated from these ratios in the main program.

ENR(I) = d_{2I}/d_{2I-2} , if ℓ-m is even,
ENR(I) = d_{2I+1}/d_{2I-1}, if ℓ-m is odd.

EX: Constant set to $10^{(NEX-5)}$. Used in testing numbers to prevent overflow on the computer.

FL: The eigenvalue approximation returned by "GETEIG".

FTERM: The largest term in the series of Eq. (12) used to form the angular function. It is used to estimate the resulting subtraction error present in the angular function.

GLIST: Array used in "CONVER" as part of the Bouwkamp procedure. It contains the γ_n^m coefficients as described in Eq. (20).

IBLIM: The number of terms used in the angular function series of Eq. (12), determined as follows: IBLIM = LIM1/2 - IX.

IM: Input parameter and the increment used to generate desired values for m.

IW6: (ℓ-m)/2, truncated to an integer.

IX: Equals zero if ℓ=m is even; equals one if ℓ=m is odd.

IXX: IX-1.

JHI: The number of Legendre functions to be calculated for a given argument, determined as follows: JHI = 2*(LNUM+CMAX+NDEC), where LMAX = C1 + (NC-1)*DC is the largest value desired for c.

L: ℓ.

LAM1-LAM5: Coefficients used in "GETEIG" to generate the power series expansion for the eigenvalue approximation in terms of c.

LIM1: 2*(L-M+C+NDEC).

LNUM: The number of successive values of ℓ starting with ℓ = m for which angular function values are desired.

M: m.

MAXAC: The number of decimal digits in the printed output for the angular functions. This parameter is set to eight in this

version of the program. See the section entitled ACCURACY OF RESULTS for information on when and how to change the parameter.

MMIN: Input parameter and starting value for m.

MNUM: Input parameter indicating the number of values of m for which angular function values are desired.

NACC: Array containing a measure of the number of decimal digits that are accurate in the printed value for the angular function.

NARG: Input parameter indicating the number of values of θ for which angular function values are desired.

NC: Input parameter indicating the number of values of c for which angular function values are desired.

NDEC: Initialization parameter that is set equal to the number of decimal digits available on the user's machine in double precision arithmetic.

NEX: Initialization parameter that is set equal to the maximum exponent size that is available on the user's machine in double precision arithmetic.

P: A doubly dimensioned array that contains the ratios of successive associated Legendre functions, where $P(k,1) = 1$, $P(k,j) = P^m_{m+j}/P^m_{m+J-1}$, with P^m_n given by Eq. (13). The index k refers to the value of θ. The special cases of $\theta = 0$, 90, and 180° are handled somewhat differently, as described in the section entitled COMPUTATIONAL PROCEDURES.

PI: Value for π, specified to 36 digits but truncated to NDEC digits by the computer.

PLEG1: Vector of length NARG containing scaling coefficients used to prevent an overflow while forming $P^m_\ell(n)$.

PNORM: Equal to $\log_{10}$ of $P^m_m(n)$ as given in Eq. (16).

PTEMP: Vector of length NARG which contains values for $P^m_\ell(n)$. These values are scaled, if necessary, to prevent computer overflow, by 10^{PLEG1}.

PTEST: Constant set to 10^{-7} degrees. Input values of θ are set equal to 0, 90, or 180° if they are within PTEST of these values.

R: ℓ-m.

RL: ℓ.

RL2: 2ℓ.

RM: m.

RM2: 2m.

S: A temporary holding array for the angular function prior to output.

SIGN: The sign of $d_{\ell-m}$.

S1: The angular function $S_{m\ell}^{(1)}(c,\eta)$.

DIMENSIONS AND STORAGE

The storage requirements for the program PANGFN are dominated by dimensioned arrays. Everything else takes about 10,000 words of storage. The minimum dimension requirement (M.D.R.) for each array is determined by the desired range of parameters as follows:

1. The M.D.R. for BLIST, GLIST, and ENR is given by (LNUM + CMAX + NDEC) where LNUM is the number of values of ℓ desired, CMAX = C1 + (NC-1) *DC is the largest value of c desired, and NDEC is the number of decimal digits in double precision words on the user's computer. This dimension is set at 250 in the listed version of PANGFN.
2. The M.D.R. for PLEG1, PNORM, and PTEMP is NARG, the number of values of θ for which angular function values are desired. This dimension is set at 10 in the listed version of PANGFN. It can be increased (or decreased) if more (or fewer) values of θ are desired.
3. The M.D.R.'s of the doubly-dimensioned array P(K,J) are NARG for K and JHI = 2*(LNUM + CMAX + NDEC) for J. The value for JHI is just twice that of the M.D.R. for BLIST, GLIST, and ENR given above in Item 1. JHI is set equal to 500 in the listed version of PANGFN.
4. All other arrays have a dimension of three. This is required for the printed output format.

If adequate computer storage is available, it is advisable to set the dimensions large enough to accommodate any anticipated parameter input and then forget about them.

PARAMETER INPUT

The input to PANGFN consists of a series of data cards as follows:

Data Card 1: Format I5 - This card contains the integer MMIN, located in the first five spaces of the card, right justified. MMIN is the smallest value of m to be used in the computation of the angular function.

Data Card 2: Format I5 - This card contains the integer IM, located in the first five spaces of the card, right justified. IM is the increment used to generate subsequent values of m from MMIN.

Data Card 3: Format I5 - This card contains the integer MNUM, located in the first five spaces of the card, right justified. MNUM is the number of values of m for which angular function values are desired.

Data Card 4: Format I5 - This card contains the integer LNUM, located in the first five spaces of the card, right justified. LNUM is the number of values of ℓ for which angular function values are desired.

Data Card 5: Format D20.10 - This card contains the value of ARG1, located in the first twenty spaces of the card. ARG1 is the initial value for θ and is used with DARG to generate all the desired values of θ.

Data Card 6: Format D20.10 - This card contains the value of DARG, located in the first twenty spaces of the card. DARG is the increment used to generate subsequent values of θ from ARG1.

Data Card 7: Format I5 - This card contains the integer NARG, located in the first five spaces of the card, right justified. NARG is the number of values of θ for which angular function values are desired.

Data Card 8: Format D20.10 - This card contains the value of C1, located in the first twenty spaces of the card. C1 is the initial value of c used with DC to generate subsequent values of c.

Data Card 9: Format D20.10 - This card contains the value of DC, located in the first twenty spaces of the card. DC is the increment used to generate subsequent values of c.

Data Card 10: Format I5 - This card contains the integer NC, located in the first five spaces of the card, right justified. NC is the number of values of c for which angular functions values are desired.

The program can easily be modified if the user desires to specify values for $\eta = \cos\theta$ rather than θ. The following changes are required:

1. Change ARG in statement 72 in the main program to BARG. Add BARG to double precision list.
2. Add statement ARG = DARCOS(ARG*PI/180.DO) immediately following statement 72 in the main program.
3. Change statement 150 in the main program to read ARGG(ISTEP) = BARG.
4. Change statment 10 in subroutine "PLEG" to read BARG = ARG.
5. Add statement ARG = DARCOS(ARG*PI/180.DO) immediately following statement 10 in "PLEG".
6. Change the print format for A(I) $\equiv \eta$ in statement 1 of subroutine "OUTPUT" from F8.3 to F8.5 to provide five digits to the right of the decimal in the printed value for η.

PARAMETER RANGES

PANGFN was developed and tested on the PDP-11/45 computer at the Underwater Sound Reference Detachment (USRD) of NRL for the following parameter ranges:

$$0° \leq \theta \leq 180°$$
$$0.00001 \leq c \leq 100$$
$$0 \leq m \leq 100$$
$$m \leq \ell \leq m+100$$

PANGFN is not limited to these ranges, however. They were chosen to be compatible with the relatively small core memory of the PDP-11/45 computer at the USRD. By increasing the dimension specifications above those given in the program listing in Appendix B to allow for more terms in the series used to calculate the angular functions, the ranges on c and ℓ-m can be increased indefinitely. The minimum dimension specifications required for larger values of c and ℓ-m are given in the section entitled DIMENSIONS AND STORAGE. There are no limitations on the range for m. Increasing m does

not require any changes in the dimension specifications. Values of c smaller than 0.00001 can be chosen, if desired. However, the output format specification for c given in statement 1005 must be changed from F15.5 to include more digits to the right of the decimal.

The version of PANGFN listed in Appendix B was also run on the Texas Instrument (TI) ASC computer at NRL and tested for the range of parameters given above. The results were consistent with those for the PDP-11/45. Since the TI ASC computer has more than 500,000 words of core memory, the dimension specification can be increased substantially on this machine. As an example, the dimensions of BLIST, GLIST, and ENR were increased to 2500 and the second dimension of P was increased to 5000 (JHI = 5000). Values of the angular function were then successfully computed for c and ℓ-m both larger than 1000.

COMPUTATIONAL PROCEDURES

There are four major computations in PANGFN: 1) determination of the eigenvalues, 2) determination of a normalizing factor for the D constants, 3) determination of the Legendre functions, and 4) calculation of the angular function.

Determination of the Eigenvalues

Accurate eigenvalues $A_{m\ell}$ are obtained by use of the variational procedure developed by Bouwkamp. This procedure, found in subroutine "CONVER", takes an approximate starting value for $A_{m\ell}$ and produces a correction $\delta A_{m\ell}$. This correction is added to the starting value to obtain a better approximation to the eigenvalue and the Bouwkamp procedure is repeated with this new approximation as the starting value. Convergence of the procedure is obtained when the relative contribution of the correction becomes less than $10^{-(NDEC-1)}$, where NDEC is the number of decimal digits used in the calculation.

The key to obtaining the correct eigenvalue lies in the choice of the initial approximation $A_{m\ell}^{(1)}$. The Bouwkamp procedure will always converge to an eigenvalue, but it will only converge to $A_{m\ell}$ when $A_{m\ell}^{(1)}$ is sufficiently close. Otherwise it will converge to another eigenvalue $A_{m\ell'}$ for the same value of m and c. The reason for this is that the Bouwkamp procedure does

not depend explicitly on ℓ but only implicitly through the eigenvalue $A_{m\ell}$. In addition, since m is restricted to even or odd values depending on whether ℓ-m is even or odd, respectively, Eq. (17) has two distinct forms--one for even ℓ-m and the other for odd ℓ-m. Therefore, the Bouwkamp procedure always converges to a characteristic value $A_{m\ell'}$ such that ℓ' has the same parity as ℓ.

The eigenvalues $A_{m\ell}$, ℓ = m, m+1, ... form a monotonically increasing sequence of positive real numbers. For each eigenvalue there exists a region of convergence $\Omega_{m\ell}$ for which the Bouwkamp procedure will converge to that eigenvalue. If the value of $A_{m\ell}^{(1)}$ is slightly greater than the upper bound of $\Omega_{m\ell}$ or slightly lower than the lower bound of $\Omega_{m\ell}$, convergence will be to $A_{m,\ell+2}$ or $A_{m,\ell-2}$, respectively.

Several different methods are used to obtain approximations to the eigenvalue for starting the Bouwkamp procedure. These include a power series expansion in c^2, an asymptotic expansion in 1/c, extrapolation of previous eigenvalues, and the approximation $A_{m\ell} = \ell(\ell+1)$. The choice of methods depends on the parameters c, m, and ℓ. Table I summarizes the choices.

Table I - Choice of method to obtain eigenvalue approximation

	c	m	Method
ℓ = m, m+1	$c \leq 6$	all	c^2 expansion
	$6 < c \leq 8$	$m < 4$	1/c expansion
	$6 < c \leq 8$	$m \geq 4$	c^2 expansion
	$c > 8$	$m < 10 + c$	1/c expansion
	$c > 8$	$m \geq 10 + c$	$\ell(\ell+1)$
ℓ = m+2, m+3	$c \leq 5$	all	c^2 expansion
	$5 < c \leq 6$	all	extrapolation
	$6 < c \leq 8$	$m < 4$	1/c expansion
	$6 < c \leq 8$	$m \geq 4$	extrapolation
	$c > 8$	$m \leq 6$	1/c expansion
	$c > 8$	$m > 6$	extrapolation
ℓ = m+4	$c \leq 8$	all	extrapolation
	$c > 8$	$m < 3$	1/c expansion
	$c > 8$	$m \geq 3$	extrapolation
$\ell > m+4$	all	all	extrapolation

This method does not, however, guarantee that $A_{m\ell}^{(1)}$ lies in $\Omega_{m\ell}$; it simply guarantees that $A_{m\ell}^{(1)}$ is somewhat close to $A_{m\ell}$. For some m, ℓ, and c the region $A_{m\ell}$ may be extremely narrow and, therefore, the estimate $A_{m\ell}^{(1)}$ would have to be very close to $A_{m\ell}$ to achieve convergence. No simple method can absolutely guarantee that $A_{m\ell}^{(1)}$ lies in $\Omega_{m\ell}$. A procedure has therefore been developed to determine if the resulting value $\dot{A}_{m\ell}$ is actually the correct eigenvalue. In order for $\dot{A}_{m\ell}$ to be consistent with the other eigenvalues previously obtained, it must satisfy the following conditions:

$$\dot{A}_{m\ell} > A_{m,\ell-1},$$

$$\dot{A}_{m\ell} - A_{m\ell}^{(1)} < A_{m\ell}^{(1)} - A_{m,\ell-1}.$$

If $\dot{A}_{m\ell}$ passes both of these tests, it is the correct eigenvalue. If $\dot{A}_{m\ell}$ fails the first inequality, it is equal to $A_{m,\ell-2}$. In this case $A_{m\ell}^{(1)}$ can be considered as a lower bound for $A_{m\ell}$. An upper bound can be determined by taking $A_{m\ell}^{(1)}$ + 1.5*CLSPAC where CLSPAC is the previous eigenvalue spacing $(A_{m,\ell-1} - A_{m,\ell-2})$.

If $\dot{A}_{m\ell}$ fails the second inequality, it is equal to $A_{m,\ell+2}$. In this case $A_{m\ell}^{(1)}$ can be considered as an upper bound for $A_{m\ell}$. A lower bound can be taken as $A_{m,\ell-1}$. Once upper and lower bounds for the eigenvalue have been established, a new starting value is taken as the mean of the bounds. The Bouwkamp procedure is then repeated with this new approximation. The resulting eigenvalue is tested to see whether it lies within the bounds. If so, $A_{m\ell}$ has been obtained and the process is concluded. If the resulting eigenvalue is less than the lower bound, then $A_{m\ell}$ must lie between the starting value and the upper bound, so that the starting value becomes the new lower bound. If the resulting eigenvalue is greater than the upper bound, then $A_{m\ell}$ must lie between the lower bound and the starting value, so that the starting value becomes the new upper bound. A new starting value is taken to be the mean of the new bounds, and the process is repeated until convergence to $A_{m\ell}$ is obtained.

Determination of the Normalizing Factor

The variational procedure of Bouwkamp includes as an intermediate step the calculation of the ratios $N_n^m = \alpha_n d_n/d_{n-2}$. Dividing by α_n as given by Eq. (22) results in the values of d_n/d_{n-2}. These ratios are well-behaved throughout the required range of n. They do not become either extremely large or extremely small. It is convenient, for computational purposes, to set the largest D constant equal to unity. The largest value occurs in the region of $n = \ell-m$, therefore, $d_{\ell-m}$ is set equal to one. In theory all of the other D constants can be obtained from $d_{\ell-m}$ by successive multiplication of the ratios d_n/d_{n-2}. In practice, however, these coefficients are not evaluated directly in PANGFN due to the computer underflow that would likely result from their extreme range in magnitude.

Angular function values obtained from Eq. (12) with $d_{\ell-m}$ set equal to unity require a normalizing factor to provide the desired Meixner-Schäfke normalization. The first step in calculating this factor is to evaluate the left-hand side of Eq. (24), starting at $n = \ell-m$ with $d_{\ell-m}$ set equal to unity and proceeding both upward and downward in n until convergence results. Each term in the series is obtained from the previous term by multiplication (upward) or division (downward) by both the square of the appropriate D constant ratio and by a ratio of integers to account for the coefficients. Convergence is assumed when the relative contribution of the last term (both upward and downward) is less than $10^{-(NDEC+1)}$, where NDEC is the number of decimal digits in double precision words in the computer. This procedure should prevent underflow or overflow during evaluation of the series. The normalizing factor DNUM is then obtained by dividing the resulting sum into the right-hand side of Eq. (24) and taking the square root. This quantity is the magnitude of the normalized value for $d_{\ell-m}$ required to provide Meixner-Schäfke normalization of the angular functions. The algebraic sign of $d_{\ell-m}$ is obtained by progressive multiplication of the signs of the ratios of the D constants, beginning with d_2/d_0 or d_3/d_1, depending on whether $\ell-m$ is even or odd, and continuing to $d_{\ell-m}/d_{\ell-m-2}$. Since d_0 and d_1 are always positive, this product of signs is equal to the sign of $d_{\ell-m}$. The angular functions are then obtained by multiplying the sum of Eq. (12) by $d_{\ell-m}$.

Determination of the Associated Legendre Functions

The associated Legendre functions required in Eq. (12) are calculated in the subroutine "PLEG". Actually, ratios of successive associated Legendre functions are calculated instead of the functions themselves because of possible overflow problems when m is large. The recursion relation of Eq. (14) is modified for this purpose to give:

$$T_n^m(\eta) \equiv P_n^m(\eta)/P_{n-1}(\eta) = \frac{2n-1}{n-m}\eta - \frac{n+m-1}{(n-m)T_{n-1}^m} , \quad n > m+1 , \tag{25}$$

with $T_m^m = 1$ and $T_{m+1}^m = (2m+1)\eta$. The ratio T_n^m is well behaved for $\eta \neq 0$; i.e., for $\theta \neq 90°$. If $\theta = 90°$, however, P_n^m is equal to zero for n-m odd, and T_n^m is unbounded or equal to zero, depending on whether n-m is even or odd. The relevant ratios in this case are those of successive nonzero values of the associated Legendre functions. These ratios are calculated using

$$T_n^m(0) = P_n^m(0)/P_{n-2}^m(0) = -(n+m-1)/n-m , \quad n-m \text{ even} , \tag{26}$$

with $T_m^m(0) = 1$.

The ratios $T_n^m(0)$ for n-m even are stored in the odd J locations of P(K,J). For convenience in calculating the successive terms in Eq. (12), the ratios $T_n^m(0)$ for n-m odd are set equal to unity and stored in the even J locations of P(K,J). To avoid possible numerical difficulties occurring near $\theta = 90°$, the program sets θ equal to 90° if it is within PTEST = 10^{-7} degrees of this value. The constant PTEST is defined near the beginning of the main program. The value of PTEST can be decreased if the user desires angular function values for θ closer than 10^{-7} degrees to 90°.

The associated Legendre functions and thus the angular functions are equal to zero when $\eta = \pm 1$; i.e., when $\theta = 0$ or 180° and m is unequal to zero. If the input value for θ is within PTEST = 10^{-7} degrees of 0 or 180° and if m is unequal to zero, the program assumes that the argument is equal to 0 or 180°, respectively, bypasses the calculation of T_m^m, and directly sets the angular function $S_{m\ell}^{(1)}(c,\pm 1)$ equal to zero.

For purposes of calculating the angular functions $S^{(1)}_{m\ell}(c,\eta)$ using the series of Eq. (12), it is convenient to set $P^m_\ell(\eta)$, the associated Legendre function that is multiplied by the D constant $d_{\ell-m}$, equal to unity, The largest term in the series is now equal to or near unity. This choice eliminates potential underflow and overflow problems in the series. The sum of the series must then be multiplied by $P^m_\ell(\eta)$ to correct for this choice. The required value for $P^m_\ell(\eta)$ is obtained from $P^m_{\ell-1}(\eta)$, which was used in the same way for $S^{(1)}_{m,\ell-1}(c,\eta)$, by multiplication by the stored ratio $P^m_\ell(\eta)/P^m_{\ell-1}(\eta)$. The value for $P^m_m(\eta)$, required for $S^{(1)}_{mm}(c,\eta)$, is calculated using Eq. (16). Since $P^m_\ell(\eta)$ can be extremely large when m is large, the value for $P^m_\ell(\eta)$ is calculated and stored as a logarithm to the base 10 to avoid overflow.

Determination of the Angular Function

The angular functions are calculated using Eq. (12). The series is evaluated by starting at $n = \ell-m$ and proceeding both upward and downward in n until convergence is obtained. Each term in the series is derived from the preceding term by use of successive multiplication (upward) or division (downward) of the ratios of D constants and the associated Legendre functions. To start the process, both $d_{\ell-m}$ and $P^m_\ell(\eta)$ and thus the initial term $d_{\ell-m}P^m_\ell(\eta)$ are set equal to unity. The logarithm to the base ten of the resulting sum is now taken and added to the corresponding logarithm of $P^m_\ell(\eta)$ and $d_{\ell-m}$. This corrects for setting both $P^m_\ell(\eta)$ and $d_{\ell-m}$ equal to unity. The resulting logarithm of the angular function is passed to the subroutine "OUTPUT". "OUTPUT" separates the mantissa from the characteristic, takes its antilogarithm, and combines it with the proper sign to obtain the base of the angular function. The number is the output in two parts: the base ($-9.9999999 \le$ base ≤ 9.9999999) and its exponent ($-999 \le$ exponent ≤ 999) as given by the characteristic. If the angular function becomes as large as 10^{1000}, possibly for very large m, or as small as 10^{-1000}, possibly for $\theta \le 10^{-7}$ degrees or $(180-\theta) \le 10^{-7}$ degrees, then the exponent printout must be appropriately expanded.

COMPUTATION TIME

The execution for PANGFN time depends on the input data. In particular, larger values of c and ℓ-m take more time. The following examples are representative of the execution times for PANGFN on the PDP-11/45 computer at USRD and the TI ASC computer at NRL for $\theta = 0°, 10°, \ldots, 90°$: $m = 0$; $\ell = 0, 1, \ldots, 100$; and selected values of c (see Table II). The execution times are nearly independent of m; e.g., the same times as shown here are obtained for $m = 100$ and $\ell = 100, 101, \ldots, 200$. The execution times where only one value of θ is desired are greater than half of those shwon above. Therefore, it is economical to include all desired values of θ in a single run. (If more than 10 values of θ are desired, some of the dimension specifications must be increased. See the section entitled DIMENSIONS AND STORAGE.)

Table II - Execution time for selected values of c

c	PDP-11/45 TIME	TI ASC TIME
1.0	42 s	2.0 s
10.0	48 s	2.4 s
50.0	70 s	3.2 s
100.0	96 s	4.3 s

PRINTED OUTPUT

The output from PANGFN consists of numerical tables, as shown in Appendix A. Numerical values for the eigenvalue $A_{m\ell}$, the desired arguments θ together with the corresponding angular functions $S_{m\ell}^{(1)}(c,\cos\theta)$, and accuracy estimates are given for desired choices of c, m, and ℓ.

The argument (ARG) is printed with three digits to the right of the decimal point. The eigenvalues (EIG) and the angular functions (S) are each printed with eight decimal digits. The accuracy estimate (ACC) is printed as an integer and indicates the number of leading decimal digits in the printed output that are likely to be accurate. A discussion of the accuracy estimate is given in the section entitled ACCURACY OF RESULTS. A procedure for changing the number of printed decimal digits in the output is outlined under the following section.

ACCURACY OF RESULTS

The procedure used to obtain the prolate eigenvalues is well-behaved numerically. Therefore all eight digits printed in the output will be accurate, with the last digit rounded.

The series expansion used to evaluate the angular function is not always well-behaved numerically. Subtraction errors can occur in the summation of the series, especially for large c and low m and ℓ. A good approximation for the number of decimal digits of accuracy lost to subtraction error in this summation is given by SE = LOG (|FTERM/S1|) where FTERM is the largest term in the series and S1 is the sum of the series. It is estimated that roundoff error, slight inaccuracies in the D constants, and the representation of the angular function as a logarithm may contribute an additional loss of two decimal digits of accuracy. Since NDEC is the number of decimal digits available for double precision words, T = NDEC-SE-2 is the expected accuracy of the calculated value of the angular function. If T is greater than eight, it is reduced to eight to correspond to the number of decimal digits printed in the output. If T is less than zero, it is set equal to zero. The resulting value for T is stored in the array NACC(K) and output under the heading ACC.

The only time that an accuracy less than eight decimal digits is likely to be encountered is when c is larger than about 30, and when m and ℓ are somewhat less than c. The subtraction error of SE digits obtained in this case corresponds to a value of $S_{m\ell}^{(1)}(c,\eta)$ near $10^{-SE}\ P_m^m(\eta)$, with the largest term in the series used to generate $S_{m\ell}^{(1)}(c,\eta)$ having a value near $P_m^m(\eta)$. In other regions of the parameters c, m, and ℓ, the value of the angular function is usually accurate to all eight digits.

If the user wishes to change the number of decimal digits printed in the output of this program from eight, he should make the following changes:

1. Change the Format statement numbered 1 in subroutine "OUTPUT" to the desired number of digits.
2. Change the Format statements numbered 1009, 1010, and 1011 in the main program to line up the headings to correspond to change 1 above.
3. Change the value of MAXAC to the desired number of printed digits.

4. Change statement number 10, the statement four lines before statement number 10, and statement number 200 in the subroutine "OUTPUT" to contain MAXAC nines (9's); i.e., as many as there are decimal digits in the output.

ACKNOWLEDGMENT

The authors gratefully acknowledge the earlier pioneering efforts by Dr. S. Hanish, Mr. R.V. Baier, and Mr. B.J. King in developing accurate and efficient computer algorithms for calculating spheroidal wave functions.

REFERENCES

1 - B.J. King and A.L. Van Buren, "A FORTRAN Computer Program for Calculating the Prolate and Oblate Angle Functions of the First Kind and Their First and Second Derivatives," NRL Report 7161 (Nov 1970).

2 - B.J. King, R.V. Baier, and S. Hanish, "A FORTRAN Computer Program for Calculating the Prolate Spheroidal Radial Functions of the First and Second Kind and Their First Derivatives," NRL Report 7012 (Mar 1970).

3 - A.L. Van Buren, R.V. Baier, and S. Hanish, "A FORTRAN Computer Program for Calculating the Oblate Spheroidal Radial Functions of the First and Second Kind and Their First Derivatives," NRL Report 6959 (Jan 1970).

4 - A.L. Van Buren, "A FORTRAN Computer Program for Calculating the Linear Prolate Functions," NRL Report 7994 (May 1976).

5 - J. Meixner and F.W. Schäfke, Mathieusche Functionen und Sphäroidfunctionen, Springer-Verlag, Berlin, Germany (1954).

6 - C. Flammer, Spheroidal Wave Functions, Standford Univ. Press, Stanford, Calif. (1957).

7 - S. Hanish, R.V. Baier, A.L. Van Buren, and B.J. King, "Tables of Radial Spheroidal Wave Functions," Vols. 1, 2, 3, Prolate, m = 0, 1, 2, respectively, NRL Reports 7088, 7089, and 7090 (1970).

8 - A.L. Van Buren, B.J. King, R.V. Baier, and S. Hanish, "Tables of Angular Spheroidal Wave Functions, Vol. 1, Prolate, m = 0," NRL Publication, U.S. Govt. Printing Office, Washington, DC (1975).

9 - S. Hanish, R.V. Baier, A.L. Van Buren, and B.J. King, "Tables of Radial Spheroidal Wave Functions," Vols. 4, 5, 6, Oblate, m = 0, 1, 2, respectively, NRL Reports 7091, 7092, and 7093 (1970).

10 - A.L. Van Buren, B.J. King, R.V. Baier, and S. Hanish, "Tables of Angular Spheroidal Wave Functions, Vol. 2, Oblate, m = 0," NRL Publication, U.S. Govt. Printing Office, Washington, DC (1975).

11 - See e.g.: E. Whittaker and G. Watson, A Course on Modern Analysis, 4th Edition, Cambridge Univ. Press, Cambridge (1952).

12 - C.J. Bouwkamp, "Theoretical and Numerical Treatment of Diffraction Through A Circular Aperture," I.E.E.E. Trans. Antennas and Propagation AP-18, 1952-176 (1970).

APPENDIX A
OUTPUT EXAMPLE

C= 10.00000 M= 5

L= 5 EIG= 0.355880860 02

ARG	S	ACC	ARG	S	ACC	ARG	S	ACC
0.000	0.0000000D+000	8	10.000	7.6451137D-003	8	20.000	3.1487679D-001	8
30.000	3.3948461D+000	8	40.000	2.0741893D+001	8	50.000	8.6942661D+001	8
60.000	2.6375871D+002	8	70.000	5.8520670D+002	8	80.000	9.4698671D+002	8
90.000	1.1125118D+003	8						

L= 6 EIG= 0.576506850 02

ARG	S	ACC	ARG	S	ACC	ARG	S	ACC
0.000	0.0000000D+000	8	10.000	1.3923460D-001	8	20.000	5.2932335D+000	8
30.000	5.0281474D+001	8	40.000	2.5905328D+002	8	50.000	8.7167687D+002	8
60.000	1.9824790D+003	8	70.000	2.9275145D+003	8	80.000	2.3651983D+003	8
90.000	0.0000000D+000	8						

L= 7 EIG= 0.796032270 02

ARG	S	ACC	ARG	S	ACC	ARG	S	ACC
0.000	0.0000000D+000	8	10.000	1.2731805D+000	8	20.000	4.4628031D+001	8
30.000	3.7158055D+002	8	40.000	1.5920220D+003	8	50.000	4.1638547D+003	8
60.000	6.5515504D+003	8	70.000	4.8516398D+003	8	80.000	-1.5294423D+003	8
90.000	-5.4603064D+003	8						

L= 8 EIG= 0.101712110 03

ARG	S	ACC	ARG	S	ACC	ARG	S	ACC
0.000	0.0000000D+000	8	10.000	7.7742883D+000	8	20.000	2.5071580D+002	8
30.000	1.8158428D+003	8	40.000	6.3396487D+003	8	50.000	1.2216896D+004	8
60.000	1.1008146D+004	8	70.000	-1.6903861D+003	8	80.000	-1.0092860D+004	8
90.000	0.0000000D+000	8						

L= 9 EIG= 0.124293780 03

ARG	S	ACC	ARG	S	ACC	ARG	S	ACC
0.000	0.0000000D+000	8	10.000	3.5727130D+001	8	20.000	1.0563102D+003	8
30.000	6.5823272D+003	8	40.000	1.8161787D+004	8	50.000	2.3303281D+004	8
60.000	4.5607642D+003	8	70.000	-1.7267797D+004	8	80.000	-2.8248982D+003	8
90.000	1.6688466D+004	8						

L= 10 EIG= 0.147678230 03

ARG	S	ACC	ARG	S	ACC	ARG	S	ACC
0.000	0.0000000D+000	8	10.000	1.3241838D+002	8	20.000	3.5712061D+003	8
30.000	1.8855714D+004	8	40.000	3.9186568D+004	8	50.000	2.6027583D+004	8
60.000	-2.0638275D+004	8	70.000	-1.7417839D+004	8	80.000	2.5314315D+004	8
90.000	0.0000000D+000	8						

*

APPENDIX B
PANGFN LISTING

```
      PROGRAM PANGFN
C
C          A FORTRAN COMPUTER PROGRAM FOR CALCULATING NUMERICAL
C     VALUES OF THE PROLATE ANGULAR FUNCTION OF THE FIRST KIND
C     AND ITS ASSOCIATED EIGENVALUES.
C
      DIMENSION BLIST(250),GLIST(250),ENR(250),P(10,500),S(3),
     1ARGG(3),IIS(3),PNORM(10),NACC(3),PLEG1(10),PTEMP(10)
      DOUBLE PRECISION AJ,ARG,ARGG,ARG1,ARR
      DOUBLE PRECISION BLIST
      DOUBLE PRECISION CL,COEF
      DOUBLE PRECISION DARG,DEC,DC,DNUM,DNEW,DOLD
      DOUBLE PRECISION EA,EIG2,EIG3,EIG4,EIG5,ENR,EX,EX1
      DOUBLE PRECISION FTERM
      DOUBLE PRECISION GLIST
      DOUBLE PRECISION C,CSQ,C1
      DOUBLE PRECISION P,PI,PLEG1,PNORM,PTEMP,PTEST
      DOUBLE PRECISION RM,RM2,RNDEC
      DOUBLE PRECISION S,S1,SIGN
      COMMON /BLK2/ BLIST,GLIST,ENR
      COMMON /BLK3/ DEC,NDEC
C
C  THE FOLLOWING TWO STATEMENTS MUST BE MODIFIED TO CONFORM WITH THE
C  USERS COMPUTER
C     NDEC: THE MAXIMUM NUMBER OF DECIMAL DIGITS AVAILABLE ON THE
C           USER'S COMPUTER IN DOUBLE PRECISION ARITHMETIC.
C     NEX:  THE MAXIMUM EXPONENT AVAILABLE FOR A DOUBLE PRECISION
C           NUMBER
C
      NDEC=16
      NEX=75
C
C
      PI=3.14159265358979323846264338327950288D0
      MAXAC=8
      RNDEC=NDEC+1
      DEC=10.D0**(-RNDEC)
      EX=10.D0**(NEX-5)
      EX1=10.D0**(5-NEX)
      PTEST=1.D-7
C
C  READ IN INITIAL ARGUMENTS
C
1     READ 1101,MMIN
      READ 1101,IM
      READ 1101,MNUM
      READ 1101,LNUM
      READ 1102,ARG1
      READ 1102,DARG
      READ 1101,NARG
*
```

```
      READ 1102,C1
      READ 1102,DC
      READ 1101,NC
C
C  BEGIN PROGRAM
C  OUTER LOOP                M LOOP
C      NEXT LOOP             C LOOP
C          INSIDE LOOP       L LOOP
C
      DO 500 IDUMM=1,MNUM
        M=MMIN+(IDUMM-1)*IM
        RM=M
        RM2=2.D0*RM
        JHI=2*(LNUM+C1+(NC-1)*DC+NDEC)
        CALL PLEG(M,JHI,ARG1,DARG,NARG,P,PNORM,PI,PTEST)
          DO 400 JNC=1,NC
          DO 3 K=1,NARG
              PTEMP(K)=1.D0
              PLEG1(K)=PNORM(K)
3           CONTINUE
            C=C1+(JNC-1)*DC
            CSQ=C*C
            EIG2=0.D0
            EIG3=0.D0
            EIG4=0.D0
            EIG5=0.D0
          PRINT 1005,C,M
          DO 300 IL=1,LNUM
            L=M+IL-1
C
C  GET THE STARTING EIGENVALUE
C
            CALL GETEIG(L,M,C,CL,EIG2,EIG3,EIG4,EIG5)
C
C  REFINE THE STARTING VALUE
C
            CALL CONVER(L,M,C,CL,EIG3,EIG5)
            IW6=(L-M)/2
            IX=L-M-2*IW6
            IXX=IX-1
            IF(L.EQ.M) GO TO 7
            DO 5 K=1,NARG
              PTEMP(K)=PTEMP(K)*P(K,L-M+1)
              IF(DABS(PTEMP(K)).LT.EX) GO TO 5
              PLEG1(K)=DLOG10(DABS(PTEMP(K)))+PLEG1(K)
              IF(PTEMP(K).LT.0.D0)PTEMP(K)=-1.D0
              IF(PTEMP(K).GT.0.D0)PTEMP(K)=1.D0
5           CONTINUE
7           CONTINUE
            LIM1=2*(L-M+C+NDEC)
*
```

```
            IBLIM=LIM1/2-IX
            SIGN=1.D0
10          DO 30 I=1,IBLIM
              ARR=IX+I+I
              EA=ARR+ARR+RM2
              IF(I.GT.IW6) GO TO 20
              SIGN=SIGN*(ENR(I)/DABS(ENR(I)))
20            ENR(I)=(EA-1.D0)*(EA+1.D0)*ENR(I)/((ARR+RM2)*
     1        (ARR+RM2-1.D0)*CSQ)
30          CONTINUE
C
C  COMPUTE THE NORMALIZING FACTOR
C
            DNUM=1.D0
            COEF=1.D0
            JLOW=L-M+2
            JTERM=IW6
            DO 50 J=JLOW,LIM1,2
              AJ=J
              JTERM=JTERM+1
              COEF=COEF*((AJ+RM2)*ENR(JTERM)/AJ)*((AJ+RM2-1.D0)*
     1        ENR(JTERM)/(AJ-1.D0))*(AJ*2.D0+RM2-3.D0)/(AJ*2.D0+RM2
     1        +1.D0)
              DNUM=DNUM+COEF
              IF(DABS(COEF/DNUM).LT.DEC) GO TO 60
50          CONTINUE
60          JLOW=L-M
            IF(JLOW.LT.2) GO TO 71
            COEF=1.D0
            JTERM=IW6
            J=JLOW
            DO 70 JJ=2,JLOW,2
              AJ=J
              COEF=COEF*(AJ/(AJ+RM2)/ENR(JTERM))*((AJ-1.D0)/(AJ+RM2
     1        -1.D0)/ENR(JTERM))*(AJ*2.D0+RM2+1.D0)/(AJ*2.D0+RM2-3.D0)
              JTERM=JTERM-1
              J=J-2
              DNUM=DNUM+COEF
              IF(DABS(COEF/DNUM).LT.DEC) GO TO 71
70          CONTINUE
71          DNUM=1.D0/DSQRT(DNUM)
            ISTEP=1
            JLOW=IW6+1
            PRINT 1006,L,CL
            IF(NARG.GE.3)PRINT 1009
            IF(NARG.EQ.2)PRINT 1010
            IF(NARG.LE.1)PRINT 1011
C
C  COMPUTE THE ANGULAR FUNCTION S
C
*
```

```
            DO 200 K=1,NARG
72             ARG=ARG1+(K-1.D0)*DARG
C
C  FOR ARG=0,90,OR 180 DEGREES USE SPECIAL METHODS TO DETERMINE
C  ANGULAR FUNCTION
C
               IF((DABS(ARG-90.D0).LT.PTEST).AND.(IX.EQ.1)) GO TO 75
               IF((DABS(ARG-180.D0).LT.PTEST).AND.(M.NE.0)) GO TO 75
               IF((DABS(ARG).LT.PTEST).AND.(M.NE.0)) GO TO 75
               GO TO 80
75             S(ISTEP)=-EX
               IIS(ISTEP)=1
               NACC(ISTEP)=MAXAC
               GO TO 150
80             FTERM=EX1
               DOLD=1.D0
               IF(PTEMP(K).LT.0.D0)DOLD=-1.D0
               S1=DOLD
               DO 100 J=JLOW,IBLIM
                 DNEW=DOLD*ENR(J)*P(K,J+J+IXX+2)*P(K,J+J+IXX+1)
                 S1=S1+DNEW
                 IF(DABS(DNEW).GT.FTERM)FTERM=DABS(DNEW)
                 IF(S1.EQ.0.D0) GO TO 95
                 IF(DABS(DNEW/S1).LT.DEC) GO TO 101
95               DOLD=DNEW
100            CONTINUE
101            IF(IW6.LT.1) GO TO 111
               DOLD=1.D0
               IF(PTEMP(K).LT.0.D0)DOLD=-1.D0
               J=IW6
               DO 110 JJ=1,IW6
                 DNEW=DOLD/(P(K,J+J+IXX+2)*P(K,J+J+IXX+1))/ENR(J)
                 S1=S1+DNEW
                 IF(DABS(DNEW).GT.FTERM)FTERM=DABS(DNEW)
                 IF(S1.EQ.0.D0) GO TO 109
                 IF(DABS(DNEW/S1).LT.DEC) GO TO 111
109              DOLD=DNEW
                 J=J-1
110            CONTINUE
111            IF(S1.EQ.0.D0) GO TO 120
               S(ISTEP)=DLOG10(DABS(S1))+DLOG10(DABS(DNUM))+
     1         DLOG10(DABS(PTEMP(K)))+PLEG1(K)
               IIS(ISTEP)=1
               IF(S1.LT.0.D0)IIS(ISTEP)=-1
               IF(SIGN.LT.0.D0)IIS(ISTEP)=-1*IIS(ISTEP)
               GO TO 125
120            S(ISTEP)=-EX
               IIS(ISTEP)=1
125            NACC(ISTEP)=DLOG10(DABS((FTERM+EX1)/(S1+EX1)))
               NACC(ISTEP)=NDEC-NACC(ISTEP)-2
*
```

```
                IF(NACC(ISTEP).LT.0)NACC(ISTEP)=0
                IF(NACC(ISTEP).GT.MAXAC)NACC(ISTEP)=MAXAC
150             ARGG(ISTEP)=ARG
C
C  TIME TO OUTPUT
C
                IF(ISTEP.NE.3) GO TO 160
                NUM=3
                CALL OUTPUT(ARGG,IIS,S,NACC,NUM)
                ISTEP=0
160             ISTEP=ISTEP+1
                IF(K.NE.NARG.OR.ISTEP.EQ.1) GO TO 200
                NUM=ISTEP-1
                CALL OUTPUT(ARGG,IIS,S,NACC,NUM)
200           CONTINUE
              PRINT 1007
300         CONTINUE
400       CONTINUE
500     CONTINUE
1005    FORMAT(43X,2HC=,F15.5,9X,2HM=,I6//)
1006    FORMAT(1X,2HL=,I6,6H  EIG=,D15.8)
1007    FORMAT(/)
1008    FORMAT(1H1,49X,23HPROLATE ANGLE FUNCTIONS/)
1009    FORMAT(3(8X,3HARG,10X,1HS,9X,3HACC,3X))
1010    FORMAT(2(8X,3HARG,10X,1HS,9X,3HACC,3X))
1011    FORMAT(8X,3HARG,10X,1HS,9X,3HACC)
1101    FORMAT(I5)
1102    FORMAT(D20.10)
9999    CONTINUE
        END
*
```

```
      SUBROUTINE PLEG(M,JHI,ARG1,DARG,NARG,P,PNORM,PI,PTEST)
      DOUBLE PRECISION API,ARG,ARG1
      DOUBLE PRECISION BARG
      DOUBLE PRECISION DARG
      DOUBLE PRECISION P,PI,PNORM,PTEST
      DOUBLE PRECISION RJ,RJ2,RM,RMP
      DIMENSION P(10,500),PNORM(10)
C
C  INITIALIZE VARIOUS COEFFICIENTS
C
      API=PI/180.D0
      RM=M
      MP=2*M-1
      DO 200 K=1,NARG
        ARG=ARG1+(K-1.D0)*DARG
        BARG=DCOS(ARG*API)
        PNORM(K)=0.D0
C
C  FOR ARG=0,90, OR 180 DEGREES USE SPECIAL METHODS TO DETERMINE
C  LEGENDRE FUNCTION RATIOS
C
        IF(DABS(ARG-90.D0).LT.PTEST) GO TO 150
        IF((DABS(ARG-180.D0).LT.PTEST).AND.(M.NE.0)) GO TO 190
        IF((DABS(ARG).LT.PTEST).AND.(M.NE.0)) GO TO 190
C
C  NORMAL COMPUTATION OF LEGENDRE FUNCTION RATIOS
C
50      P(K,1)=1.D0
        P(K,2)=(2.D0*RM+1.D0)*BARG
        DO 100 J=3,JHI
          RJ=J+RM
          RJ2=2.D0*RJ
          P(K,J)=((RJ2-3.D0)*BARG-(RJ+RM-2.D0)/P(K,J-1))/(RJ-RM-1.D0)
100     CONTINUE
        IF(M.EQ.0) GO TO 200
        IM=MP
        DO 120 IIM=1,MP,2
          RMP=IM
          PNORM(K)=PNORM(K)+DLOG10(RMP)+DLOG10(DABS(DSIN(ARG*API)))
          IM=IM-2
120     CONTINUE
        GO TO 200
C
C  COMPUTATION OF LEGENDRE FUNCTION RATIOS FOR ARG=90 DEGREES
C
150     P(K,1)=1.D0
        DO 160 J=3,JHI,2
          RJ=J+RM
          P(K,J)=-(RJ+RM-2.D0)/(RJ-RM-1.D0)
          P(K,J-1)=1.D0
*
```

```
160       CONTINUE
          IF(M.EQ.0) GO TO 200
          IM=MP
          DO 170 IIM=1,MP,2
            RMP=IM
            PNORM(K)=PNORM(K)+DLOG10(RMP)
            IM=IM-2
170       CONTINUE
          GO TO 200
190       DO 195 J=1,JHI
195       P(K,J)=0.D0
200     CONTINUE
        RETURN
        END
*
```

```
      SUBROUTINE GETEIG(L,M,C,CL,EIG2,EIG3,EIG4,EIG5)
      DOUBLE PRECISION BOSH
      DOUBLE PRECISION CL,C,CSQ
      DOUBLE PRECISION EIG2,EIG3,EIG4,EIG5
      DOUBLE PRECISION R,RL,RL2,RM,RM2
      DOUBLE PRECISION LAM1,LAM2,LAM3,LAM4
C
C  COMPUTE SOME HEAVILY USED VARIABLES
C
      CSQ=C*C
      RL=L
      RL2=2.D0*RL
      RM=M
      RM2=2.D0*RM
      R=RL-RM
C
C  THE CASE OF SMALL M AND LARGE C
C
      IF(L.EQ.(M+4).AND.C.GT.8.D0.AND.M.LT.3) GO TO 200
C
C  APPROXIMATIONS BASED ON PREVIOUS EIGENVALUES, IF AVAILABLE
C
      IF(L.GT.(M+3)) GO TO 30
C
C USE EXPANSION IN TERMS OF CSQ FOR LOW C, AND THE EXPANSION
C IN TERMS OF 1/C FOR LARGE C (C>8)
C
      IF(C.GT.8.D0) GO TO 100
      IF(C.GT.6.D0.AND.M.LT.4) GO TO 200
C
C  COMPUTE COEFFICIENTS FOR CSQ EXPANSION
C
      IF(L.GT.(M+1).AND.C.GT.5.D0) GO TO 30
      LAM1=RL*(RL+1.D0)
      LAM2=.5*(1.-(RM2-1.)*(RM2+1.)/((RL2-1.)*(RL2+3.)))
      LAM3=.5*(RL-RM-1.)*(RL-RM)*(RL+RM-1.)*(RL+RM)/
     1((RL2-3.)*(RL2-1.)*(RL2-1.)*(RL2-1.)*(RL2+1.))-
     2.5*(RL-RM+1.)*(RL-RM+2.)*(RL+RM+1.)*(RL+RM+2.)/
     3((RL2+1.)*(RL2+3.)**3*(RL2+5.))
      LAM4=(4.*RM*RM-1.)*((RL-RM+1.)*(RL-RM+2.)*(RL+RM+1.)*(RL+RM+2.)
     1/((RL2-1.)*(RL2+1.)*(RL2+3.)**5*(RL2+5.)*(RL2+7.))-
     2(RL-RM-1.)*(RL-RM)*(RL+RM-1.)*(RL+RM)
     3/((RL2-5.)*(RL2-3.)*(RL2-1.)**5*(RL2+1.)*(RL2+3.)))
      CL=LAM1+CSQ*(LAM2+CSQ*(LAM3+LAM4*CSQ))
      EIG2=EIG3
      EIG3=EIG4
      EIG4=EIG5
      RETURN
30    CONTINUE
C
*
```

```
C  EXPANSIONS BASED ON PREVIOUS EIGENVALUES
C  FIRST THROUGH THIRD ORDER
C
      IF(L.GT.(3+M)) GO TO 50
      IF(L.GT.(2+M)) GO TO 40
C
C  FIRST ORDER
C
      CL=2.D0*EIG5-EIG4
31    EIG3=EIG4
      EIG4=EIG5
      RETURN
C
C  SECOND ORDER
C
40    CL=3.D0*(EIG5-EIG4)+EIG3
41    EIG2=EIG3
      GO TO 31
C
C  THIRD ORDER
C
50    CL=4.D0*(EIG3+EIG5)-6.D0*EIG4-EIG2
      GO TO 41
100   CONTINUE
      IC=C
C
C  IF M>6 THEN THE EIGENVALUES ARE VERY REGULARLY SPACED
C
      IF(M.GT.6.AND.L.GT.M+1) GO TO 30
C
C  USE L*(L+1) APPROXIMATIONS FOR L=M AND L=M+1
C
      IF(M.LT.(10+IC)) GO TO 200
      CL=RL*(RL+1.D0)
      EIG4=EIG5
      RETURN
C
C  COMPUTE ESTIMATE WITH ASYMPTOTIC EXPANSION
C
200   BOSH=1.D0
      IF(M.EQ.0) BOSH=0.D0
      CL=(R+R+1.)*C-.25*(2.*R*R+R+R+3.)-(R+R+1.)*(R*R+3.)/(C*16.)
     1+RM*RM+(RM-1.)*(RL-RM)*BOSH
      EIG2=EIG3
      EIG3=EIG4
      EIG4=EIG5
      RETURN
      END
*
```

```
      SUBROUTINE CONVER(L,M,C,CL,EIG3,EIG5)
      DOUBLE PRECISION BLIST
      DOUBLE PRECISION CL,CLL,CLSPAC,CLU,CORA,CORB
      DOUBLE PRECISION DE,DEC,DL
      DOUBLE PRECISION EIG3,EIG5,ENR,ENRC
      DOUBLE PRECISION FL
      DOUBLE PRECISION GL,GLIST
      DOUBLE PRECISION C,CSQ
      DOUBLE PRECISION RM,RM2,R
      DIMENSION BLIST(250),GLIST(250),ENR(250)
      COMMON /BLK2/ BLIST,GLIST,ENR
      COMMON /BLK3/ DEC,NDEC
C
C  CALCULATE SOME HEAVILY USED CONSTANTS
C
      CSQ=C*C
      RM=M
      RM2=2.D0*RM
C
C  DETERMINE EIGENVALUE SPACING OF PREVIOUS EIGENVALUES
C
      GL=EIG5
      IF(L.EQ.M) CLSPAC=CL
      IF(L.NE.M) CLSPAC=EIG5-EIG3
      IW6=(L-M)/2
C
C  IF EST<LAST EIG THEN EST=LAST EIG
C
      IF(CL.LT.GL)CL=GL
      FL=CL
      JNDE=0
      IX=L-M-2*IW6
      ISC=2+IX
      LIM1=2*(L-M+C+NDEC)
      J=1
      IF(LIM1.LT.ISC) GO TO 25
C
C  COMPUTE BETA COEF.
C
      DO 20 I=ISC,LIM1,2
        R=I
        BLIST(J)=R*(R-1.D0)*(RM2+R)*(RM2+R-1.D0)
     1  *CSQ*CSQ/((RM2+2.D0*R-1.D0)
     2  *(RM2+2.D0*R-3.D0)*(RM2+2.D0*R+1.D0))
        J=J+1
20    CONTINUE
25    J=1
      ID21=ISC-1
      LIM11=LIM1+1
      IF(LIM11.LT.ID21) GO TO 35
*
```

```
C
C  COMPUTE THE GAMMA COEF.
C
      DO 30 I=ID21,LIM11,2
        R=I-1
        GLIST(J)=(RM+R)*(RM+R+1.D0)+.5D0*CSQ*(1.D0
     1  -(4.D0*RM*RM-1.D0)/((RM2+
     2  2.D0*R-1.D0)*(2.D0*RM+2.D0*R+3.D0)))
        J=J+1
30    CONTINUE
35    IFC=1
      IBLIM=LIM1/2-IX
      IGLIM=IBLIM+1
      IRIO=IW6+1
      IW1=IW6+2
40    ENR(1)=CL-GLIST(1)
      IF(IW6.LT.1) GO TO 55
C
C  EVALUATE THE CONTINUED FRACTION
C
      DO 50 I=1,IW6
        ENR(I+1)=-BLIST(I)/ENR(I)-GLIST(I+1)+CL
50    CONTINUE
55    ENR(IBLIM)=-BLIST(IBLIM)/(GLIST(IGLIM)-CL)
      IW15=IBLIM-1
      IP=IW1+IW15
      IF(IW15.LT.IW1) GO TO 65
C
C  EVALUATE THE CONTINUED FRACTION
C
      DO 60 I=IW1,IW15
        IPI=IP-I
        ENR(IPI)=-BLIST(IPI)/(GLIST(IPI+1)-CL+ENR(IPI+1))
60    CONTINUE
65    ENRC=-BLIST(IRIO)/(GLIST(IRIO+1)-CL+ENR(IRIO+1))
      DE=ENRC*ENRC/BLIST(IRIO)
      CORB=DE
      IF(IBLIM.LT.IW1) GO TO 75
C
C  COMPUTE THE DENOMINATOR IN THE BOUWKAMP
C
      DO 70 I=IW1,IBLIM
        DE=ENR(I)*ENR(I)/BLIST(I)*DE
        CORB=CORB+DE
        IF (DABS(DE/CORB).LT.DEC) GO TO 75
70    CONTINUE
75    CORA=1.D0
      DE=1.D0
      IF(IW6.LT.1) GO TO 90
C
*
```

```
C  COMPUTE THE DENOMINATOR IN THE BOUWKAMP
C
      DO 80 I=1,IW6
        DE=BLIST(IRIO-I)/(ENR(IRIO-I)*ENR(IRIO-I))*DE
        CORA=CORA+DE
        IF(DABS(DE/CORA).LT.DEC) GO TO 90
80    CONTINUE
C
C  COMPUTE THE CORRECTION TO THE EIGENVALUE
C
90    DL=(ENRC-ENR(IRIO))/(CORA+CORB)
      CL=CL+DL
C
C  EIGENVALUE ACCURATE ENOUGH?
C
      IF(DABS(DL/CL).LT.DEC) GO TO 100
      IFC=IFC+1
      IF(IFC.LT.NDEC) GO TO 40
100   CONTINUE
C
C  IF RANGE OF EIGENVALUE ALREADY ESTABLISHED (JNDE NE 0) BRANCH
C  TO TEST WHETHER THE RESULTING EIGENVALUE IS WITHIN THE
C  ESTABLISHED RANGE
C
      IF(JNDE.NE.0) GO TO 120
C
C  TEST OF THE EIGENVALUE FOR JNDE = 0
C
      IF(CL.GT.GL) GO TO 115
C
C  CONVERGED TO THE NEXT LOWER EIGENVALUE OF THE SAME PARITY
C
      CLL=FL
      CLU=FL+CLSPAC*1.5D0
      GO TO 130
115   IF((CL-FL).LT.(FL-GL)) GO TO 140
C
C  CONVERGED TO THE NEXT HIGHER EIGENVALUE OF THE SAME PARITY
C
      CLU=FL
      CLL=.5D0*(FL+GL)
      GO TO 130
C
C  NARROWING OF RANGE IF THE CORRECT EIGENVALUE HAS NOT BEEN OBTAINED
C
120   IF(CL.GT.CLL) GO TO 122
      CLL=FL
      GO TO 130
122   IF(CL.LT.CLU) GO TO 140
      CLU=FL
*
```

```
C
C  EIGENVALUE IS NOW SOMEWHERE IN THE RANGE ESTABLISHED ABOVE
C  CHOOSE THE MIDPOINT AND REPEAT THE BOUWKAMP PROCEDURE
C
130    CL=.5D0*(CLL+CLU)
       FL=CL
       IFC=1
       JNDE=JNDE+1
C
C  IF MORE THAN 50 NARROWINGS ARE REQUIRED THEN SOMETHING IS WRONG
C
       IF(JNDE.EQ.50) GO TO 900
       GO TO 40
140    EIG5=CL
       RETURN
C
C  ERROR PRINTOUT
C
900    PRINT 999,L
999    FORMAT(1X,37HERROR IN EIGENVALUE ROUTINE CONVER AT/
      113H EIGENVALUE #,I5,29H THIS VALUE MAY BE INACCURATE)
       RETURN
       END
*
```

```
      SUBROUTINE OUTPUT(A,IB,B,NACC,NUM)
      DIMENSION A(3),IB(3),B(3),DIB(3),DB(3),IIB(3),IP1(3),IP2(3),
     1IP3(3),ISIG(3),NACC(3)
      INTEGER PLUS,MINUS
      DATA PLUS/1H+/,MINUS/1H-/
      DOUBLE PRECISION A,B,DIB,DB
      DO 50 I=1,NUM
    2   IF(B(I).LE.-999.D0) GO TO 100
    3   IF(B(I).GE.999.D0) GO TO 200
        DIB(I)=IB(I)
        IIB(I)=B(I)
        DB(I)=IIB(I)
        B(I)=B(I)-DB(I)
        B(I)=10.D0**B(I)
        IF(B(I).GT..99999999D0) GO TO 10
        B(I)=B(I)*10.D0
        IIB(I)=IIB(I)-1
        GO TO 20
   10   IF(B(I).LT.9.9999999D0) GO TO 20
        B(I)=B(I)/10.D0


        IIB(I)=IIB(I)+1
   20   B(I)=B(I)*DIB(I)
        ISIG(I)=PLUS
        IF(IIB(I).GE.0) GO TO 30
        IIB(I)=-1*IIB(I)
        ISIG(I)=MINUS
   30   IP1(I)=IIB(I)/100
        IP2(I)=IIB(I)/10-IP1(I)*10
        IP3(I)=IIB(I)-IP1(I)*100-IP2(I)*10
   50 CONTINUE
      PRINT 1,(A(I),B(I),ISIG(I),IP1(I),IP2(I),IP3(I),NACC(I),I=1,NUM)
      RETURN
  100 B(I)=0.D0
      ISIG(I)=PLUS
      IP1(I)=0
      IP2(I)=0
      IP3(I)=0
      GO TO 50
  200 B(I)=9.9999999D0
      ISIG(I)=PLUS
      IP1(I)=9
      IP2(I)=9
      IP3(I)=9
      NACC(I)=0
      GO TO 50
    1 FORMAT(3(5X,F8.3,1X,F10.7,1HD,A1,3I1,3X,I1,4X))
      END
*
```